Constructing Optimal Axis-Parallel Highways

Heekap Ahn Tetsuo Asano Sang Won Bae
Otfried Cheong Chan-Su Shin Alexander Wolff

September 12, 2005

Abstract

In this paper we consider the problem of constructing optimal highways. For two points p and q in the plane, a line h—the highway—and a real $v > 1$, we define the *travel time* (also known as the *City distance*) from p and q to be the time needed to traverse a quickest path from p to q, where distances are measured in the Manhattan (i.e. the L_1-) metric, and the speed on h is v and elsewhere 1. Given a set S of n points in the plane and a highway speed v, we show how to find an axis-parallel line that minimizes the maximum travel time over all pairs of vertices. Our algorithms takes $O(n \log n)$ time.

We also show that placing k parallel highways does not reduce the maximum travel time.

1 Introduction

2 Our model

We give an algorithm that computes an optimal vertical highway. Clearly the same algorithm can also be used to find an optimal horizontal highway. We decided to describe the vertical case since it allows us to embed the points and their travel-time graphs (defined below) into the same plane.

For two points p and q in the plane and a vertical highway h_x at x-coordinate x with speed $v > 1$, we define the *highway distance* $\mathrm{hw}_x(p, q)$ of $p(x_p, y_p)$ and $q(x_q, y_q)$ to be the time needed to traverse a quickest path from p via h_x to q. Note that

$$\mathrm{hw}_x(p, q) = |x_p - x| + |y_p - y_q|/v + |x - x_q|.$$

Then the travel time $t_{pq}(x)$ from p to q in the presence of highway h_x is the minimum of the L_1-distance and the highway distance of p and q. Given a set S of n points in the plane, our goal is to find a vertical highway that minimizes the maximum travel time over all pairs of points in S. Note that the graph of the function t_{pq} that maps the x-coordinate of a vertical highway h_x to the travel time from p to q has a very simple structure. If we let $v' = (v-1)/(2v)$, $\Delta y = |y_q - y_p|$, and assume $x_p \leq x_q$ then

$$l_{pq} = x_p - \Delta y \cdot v' \quad \text{and} \quad r_{pq} = x_q + \Delta y \cdot v'$$

are the two x-coordinates where $L_1(p, q) = \mathrm{hw}_x(p, q)$. The travel-time graph Γ_{pq} of t_{pq} (see Figure 1) consists of three horizontal segments

$$s^1_{pq} = (-\infty, l_{pq}] \times L_1(p, q), \quad s^3_{pq} = [x_p, x_q] \times \mathrm{hw}_x(p, q), \quad s^5_{pq} = [r_{pq}, \infty) \times L_1(p, q)$$

and two line segments s^2_{pq} and s^4_{pq} that connect the segment s^1_{pq} to s^3_{pq} and the segment s^3_{pq} to s^5_{pq}, respectively. The slopes 2 and -2 of the non-horizontal segments s^2_{pq} and s^4_{pq}, respectively, do not depend on the highway speed v. We refer to s^3_{pq} as the *valley floor* and to the two other horizontal segments $^1_{pq}$ and s^5_{pq} as *plateaus* of the travel-time graph Γ_{pq} or simply of $\{p, q\}$.

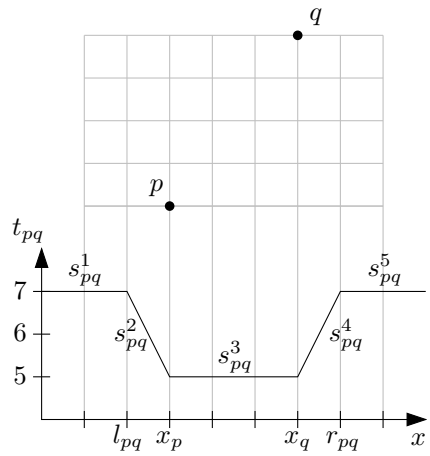


Figure 1: The travel time t_{pq} as a function of the x-coordinate of the highway.

Now consider the travel-time graphs Γ_{pq} of all pairs $\{p, q\} \subseteq S$. Their upper envelope $\mathcal{E}$ maps x to the maximum travel time over all pairs $\{p, q\}$. Thus a global minimum of $\mathcal{E}$ corresponds to a highway position that minimizes the maximum travel time. Let x^* be the leftmost such highway position. It is clear that x^* is bounded if not all points lie on the same horizontal line. Let h^* be the highway with x-coordinate x^* and let $t^* = \max_{\{p,q\} \subseteq S} t_{pq}(x^*)$ be the maximum travel time given h^*. Our goal is to compute x^* and t^* efficiently.

It is easy to determine (x^*, t^*) in $O(n^2 \log n)$ time as follows. Clearly two travel-time graphs cross each other only a constant number of times. For each crossing consider the leftmost intersection point. There are $O(n^2)$ such intersection points in total. A plane sweep that stops at each of these intersection points and maintains $\mathcal{E}$ in $O(\log n)$ time per intersection yields (x^*, t^*) within the desired time bound. In the next section we characterize the optimal highway position. Based on this characterization we then give an algorithm that determines (x^*, t^*) in $O(n \log n)$ time.

In the remainder of the paper we assume that S conatins at least three points and that not all points have the same y-coordinate.

3 Characterization of the optimum

Lemma 1 *The upper envelope $\mathcal{E}$ is concave.*

Proof. All travel-time graphs are concave, thus their point-wise maximum is concave, too. $\square$

Recall that x^* is the leftmost highway position that minimizes the maximum travel time over all pairs of points in S. In the sequel we will abbreviate $t_{pq}(x^*)$ by t^*_{pq}.

Lemma 2 *There are two pairs $\{a,b\}$ and $\{c,d\}$ in S with $x_a \le x^* \le x_b$ and $x^* \le x_c \le x_d$ whose highway distances given h^* equal t^*, i.e.* $\mathrm{hw}_{x^*}(a,b) = \mathrm{hw}_{x^*}(c,d) = t^*$. *In particular,* $(x^*, t^*) = s^2_{cd} \cap s^3_{ab}$.

Proof. Since we are looking for a *leftmost* minimum (x^*, t^*), it must lie on the intersection of two travel-time graphs, and one of the line segments participating in the intersection must be of type s^2. This is due to the fact that segments of type s^2 are the only pieces of travel-time graphs that go down. Let $\{c,d\}$ be the pair with $(x^*, y^*) \in s^2_{cd}$. This immediately yields that $x^* \le x_c \le x_d$. We have three cases. In case (A) the other segment that participates in the intersection is of type s^4_{pq} and in case (B) the other segment is a plateau of some graph Γ_{pq}. In both cases we show that we are actually also in case (C), i.e. there is a pair $\{a,b\}$ of points in S whose valley floor contains the minimum. This yields $x_a \le x^* \le x_b$. We first consider case (A).

In case (A), the optimal highway h^* separates the pairs $\{p,q\}$ and $\{c,d\}$, and the highway distances of both pairs are equal to the maximum travel time t^*. Let Π_{pq} and Π_{ab} be the two shortest paths that connect p to q and c to d via the highway h^*, respectively. Let $I = \Pi_{pq} \cap \Pi_{cd}$. We first consider the subcase $I \ne \emptyset$, see Figure 2.

For $r \in \{p,q,c,d\}$ let ℓ_r be the travel time from r to the closest point of I. Without loss of generality we assume $y_q < y_p$ and $y_d < y_c$. Suppose $\ell_d > \ell_q$. This would mean that $t^*_{pd} > t^*_{pq}$, which in turn would contradict the maximality of t^*_{pq}. By a symmetric argument we can rule out $\ell_d < \ell_q$. Thus $\ell_d = \ell_q$, which means that $t^*_{pd} = t^*_{pq}$ and that t^* lies in the valley floor of $\{p,d\}$.

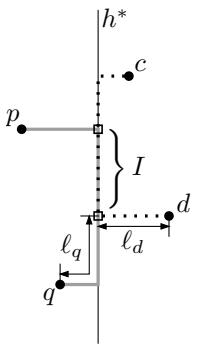


Figure 2: A tie with $\Pi_{pq} \cap \Pi_{cd} \ne \emptyset$.

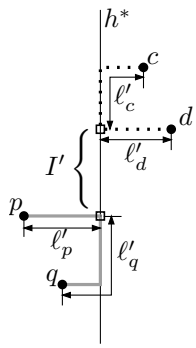


Figure 3: A tie with $\Pi_{pq} \cap \Pi_{cd} = \emptyset$.

Now we consider the subcase $I = \emptyset$, see Figure 3. Note that in this subcase $h^* \setminus (\Pi_{pq} \cup \Pi_{cd})$ has three connected components. Let I' be the middle, i.e. the bounded component, including the two boundary points. For $r \in \{p,q,c,d\}$ let ℓ'_r be the travel time from r to the point of I' that is closest to r. Note that

$$t^*_{pq} = \ell'_p + \ell'_q = \ell'_c + \ell'_d = t^*_{cd} = t^*.$$

Due to the maximality of t^*_{pq} we have $t^*_{pq} \ge t^*_{pc}$, t^*_{pd}, t^*_{qc}, t^*_{qd}. Summing up these four inequalities yields

$$\ell'_p + \ell'_q = \ell'_c + \ell'_d + 2|I'|/v,$$

where $|I'|$ is the length of the line segment I'. Now it is clear that I' has length 0 and that we can proceed as in the subcase $I \neq \emptyset$.

In case (B) the travel time from p to q is determined by the L_1-distance of p and q, i.e. $t^*_{pq} = L_1(p,q) \leq \mathrm{hw}_{x^*}(p,q)$. Given that c and d lie to the right of h^*, p and q must lie to the left of h^*, otherwise we could move the highway to the right and thus closer to both pairs. This would decrease the travel time from c to d without increasing the travel time from p to q, contradicting the minimality of t^*. If h^* separates the two pairs, then we must have $t^*_{pq} = \mathrm{hw}_{x^*}(p,q)$ otherwise we could again move the highway to the right, contradicting the minimality of t^*. Thus we are actually in case (A), where the travel time of both pairs is determined by their highway distance. $\square$

It should be possible to simplify the above case analysis. Any suggestions?

4 Algorithm

Due to Lemma 2 we know that the minimum of $\mathcal{E}$ lies on the intersection of a segment of type s^2 and a segment of type s^3, i.e. there are pairs $\{a,b\}$ and $\{c,d\}$ in S such that $(x^*,t^*) = s^2_{cd} \cap s^3_{ab}$. The first task is

(A) go through all (combinatorially different) highway positions. For each such highway position x we find a pair $\{\alpha_x, \beta_x\}$ in S of maximum highway distance among all pairs in S that are separated by a highway at x.

Note that $\{a,b\} = \arg\min_x \mathrm{hw}_x(\alpha_x, \beta_x)$, the pair that has minimum highway distance among all pairs of type $\{\alpha_x, \beta_x\}$, is the first of the two point pairs we are looking for. **State reason!**

We do a binary search on the list $L = \{l_{pq} \mid p \neq q \in S,\ x_p \leq x_q\}$. For each step in the search we call a decision algorithm. The second task is

(B) find a pair $\{\gamma, \delta\}$ in S that is not separated by the current highway and whose travel-time distance is larger than the highway distance of a and b if such a pair exists.

If a pair $\{\gamma, \delta\}$ exists, then the highway must be moved into the direction of γ and δ, since this is the only way to decrease their travel-time distance. If no pair exists whose travel-time distance is larger than $\mathrm{hw}_x(a,b)$, then by Lemma 2 there must be a pair $\{c,d\}$ on the right side of the current highway with $\mathrm{hw}_x(c,d) = \mathrm{hw}_x(a,b)$. In this case the current highway position is optimal. This is how we can decide whether we have found the optimal highway position or whether we have to continue our search in the upper or in the lower part of the list.

4.1 Binary search

In order to avoid sorting the list L, which has $\Theta(n^2)$ elements, we use the fact that each value ℓ_{pq} can be written as the sum of two terms such that one term depends only on p and the other only on q:

$$\ell_{pq} = \begin{cases} (x_p + y_p v') - y_p v' & \text{if } y_q \geq y_p \\ (x_p - y_p v') + y_p v' & \text{else.} \end{cases}$$

Recall that we required $x_p \leq x_q$ for all $l_{pq} \in L$. After sorting the lists $L_1 = \{x_p \pm y_p v' \mid p \in S\}$ and $L_2 = \{\pm y_q v' \mid q \in S\}$ we can determine the k-th largest value in the list $L_{1\times 2} = \{v_1 + v_2 \mid v_1 \in L_1, v_2 \in L_2\}$ in $O(n)$ time using the method of **REF?**. Note that $L \subsetneq L_{1\times 2}$. Only a fraction of roughly 1/4 of the elements in $L_{1\times 2}$ is contained in L, but this does not matter neither for the correctness of our method nor for its asymptotic running time.

Due to our decision algorithm it seems we do not need the concavity of $\mathcal{E}$.

4.2 Decision algorithm

In this section we show the following:

Theorem 1 *After an $O(n \log n)$-time preprocessing we can answer queries of the following type in linear time: given $x \in \mathbb{R}$, decide whether $x < x^*$, $x = x^*$, or $x > x^*$ (i.e. the sign of $x - x^*$).*

We do this by solving the two tasks (A) and (B) mentioned in the beginning of this section. We first tackle task (A) and observe the following.

Observation 1 *No point pair separated by a vertical highway h_x changes its highway distance while h_x is moved to the right without hitting an input point.*

Consider a point pair $\{\alpha, \beta\}$ that has maximum highway distance among all pairs in S that are separated by a highway at a given x-coordinate x. We give an algorithm that efficiently determines such pairs for all x-coordinates of input points. For the description of our algorithm we need the following notation. Let $\mathcal{L}^{\mathrm{ll}}$ and $\mathcal{L}^{\mathrm{ul}}$ be the sets of straight lines of slopes v and $-v$, respectively, that go through input points p with $x_p \leq x$, see Figure 4. For input points q with $x_q \geq x$, define $\mathcal{L}^{\mathrm{ur}}$ and $\mathcal{L}^{\mathrm{lr}}$ analogously. Let g^{ll} and g^{lr} be the bottommost lines in $\mathcal{L}^{\mathrm{ul}}$ and $\mathcal{L}^{\mathrm{ur}}$, respectively. Analogously, let g^{ul} and g^{ur} be the topmost lines in $\mathcal{L}^{\mathrm{ll}}$ and $\mathcal{L}^{\mathrm{lr}}$, respectively. Clearly, these lines and sets of lines depend on the position x of the highway, but instead of indexing all these variables additionally by x, we will always make sure that it is clear to which highway the variables refer. Let h_x^- (h_x^+) be the closed halfplane bounded to the right (left) by h_x.

We now characterize point pairs that have maximum highway distance among those pairs separated by the highway.

Lemma 3 *Let $\{\alpha, \beta\}$ with $x_\alpha \leq x_\beta$ be a pair of maximum highway distance among the pairs that are separated by h_x. Then $\alpha \in g^{\mathrm{ul}}$ and $\beta \in g^{\mathrm{lr}}$, or $\alpha \in g^{\mathrm{ll}}$ and $\beta \in g^{\mathrm{ur}}$.*

Proof. Let $\{\alpha, \beta\}$ be as required. Our proof is by contradiction. We assume that one of the two points lies on neither of the two lines, say α lies neither on g^{ul} nor on g^{ll}, see Figure 5. Let π be the shortest a–b path in the highway metric and let β' be the point on h_x closest to β. Note that β' and β have the same y-coordinate and that *all* highway paths from β to points in h_x^- go through β'. Consider the set W_δ of points in h_x^- that have the same distance $\delta > 0$ from β' in the highway metric. Note that this so-called *wavefront* W_ρ forms a triangle of width ρ and height $2\rho v$. The triangle is bounded by h_x, one line segment of slope v and one of slope $-v$. A wavefront $W_{\rho'}$ with $\rho' > \rho$ is a scaled copy of W_ρ with scaling factor ρ'/ρ and scaling center β'. When the wavefront hits α, the closure of the wavefront contains all points in h_x^- whose distance from β is bounded by $\mathrm{hw}_x(\alpha, \beta')$. However, due to

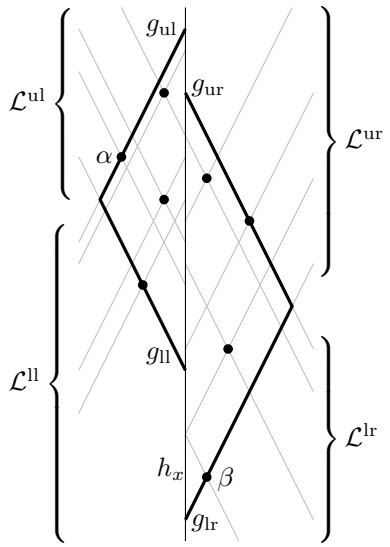


Figure 4: Extremal pairs for $v = 2$.

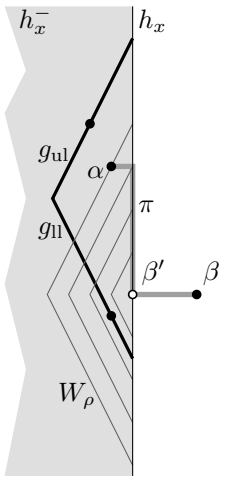


Figure 5: Wavefront W_ρ touches α before g^{ul}.

our assumption, at least one of the lines g^{ul} or g^{ll} has not yet been touched by the wavefront. Thus any point that defines the untouched line is farther from β' than α, and hence farther from β, too. This contradicts the definition of α. ◻

Next we show how to determine a pair of maximum highway distance given points that fulfill the characterization of Lemma 3. Let $S_x^- = S \cap h_x^-$ and analogously $S_x^+ = S \cap h_x^+$.

Lemma 4 *Given two points in S_x^-, one on g^{ul} and one on g^{ll}, as well as two points in S_x^+, one on g^{ur} and one on g^{lr}, we can determine in constant time a pair in S that has maximum highway distance among all pairs separated by h_x.*

Proof. Let y^- be the y-coordinate of the intersection point of g^{ul} and g^{ll}. Define y^+ accordingly using g^{ur} and g^{lr}. If $y^- > y^+$ (as in Figure 4), then all pairs that consist of a point on g^{ul} and a point on g^{lr} have the same highway distance. This can be seen by observing that every shortest path connecting such a pair (q, r) consists of a piece from q to the point $p^-(x, y^-) \in h_x$, a piece exclusively on h_x, and a piece from $p^+(x, y^+) \in h_x$ to r. Note that the first and the third piece are disjoint. Due to the slopes of g^{ul} and g^{lr}, all points in $g^{\mathrm{ul}} \cap h_x^-$ have the same highway distance from p^- and all points in $g^{\mathrm{lr}} \cap h_x^+$ have the same highway distance ρ from p^+.

On the other hand all pairs that consist of a point on g^{ll} and a point on g^{ur} have highway distance strictly less than ρ. This is due to the fact that—other than above—a shortest path from a point in $g^{\mathrm{ll}} \cap h^-$ to p^- and a shortest path from a point in $g^{\mathrm{ur}} \cap h^+$ to p^+ overlap.

Clearly the case $y^- < y^+$ is symmetric. If $y^- = y^+$, each point pair in $g^{\mathrm{ul}} \cap h_x^- \times g^{\mathrm{lr}} \cap h_x^+$ has the same highway distance as a point pair in $g^{\mathrm{ll}} \cap h_x^- \times g^{\mathrm{ur}} \cap h_x^+$, since all points in these sets have the same highway distance to the point p^+, which in this case coincides with the point p^-. ◻

We solve task (A) as follows. We sort the points according to ascending x-coordinate. We first scan the resulting array X in the given order and determine for each coordinate $x \in X$ the topmost line g^{ll} with slope v and the bottommost line g^{ul} with slope $-v$ among the lines through points q with $x_q \leq x$. Now we scan X in reverse order and determine for each $x \in X$ the topmost line g^{lr} with slope $-v$ and the bottommost line g^{ur} with slope v among the lines through points q with $x_q \geq x_p$. With each of the lines we store a point that defined the line.

Then we scan X a third time and for each value $x \in X$ we consider the points on $\{g^{\mathrm{ul}}, g^{\mathrm{lr}}\}$ and on $\{g^{\mathrm{ll}}, g^{\mathrm{ur}}\}$. Among them by Lemma 4 we can find in constant time the desired point pair $\{\alpha_x, \beta_x\}$ of maximum highway distance (given a highway at x). After sorting the three scans can be done in $O(n)$ time.

Now, by Lemma 2 we know that t^* is determined by the highway distance of a point pair $\{a, b\}$ separated by h^*. Clearly $\{a, b\}$ must be a pair whose highway distance is maximum among the pairs separated by h^*. Due to Observation 1 we can restrict our search for such pairs to highways that go through input points. Thus we already know the value of t^*, namely

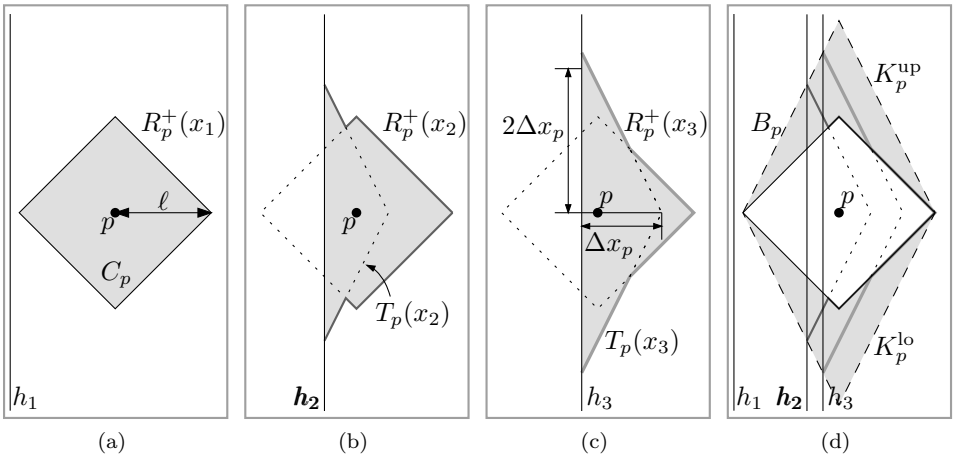
$$t^* = \min_{x \in X} \mathrm{hw}_x(\alpha_x, \beta_x), \tag{1}$$

but we still do not know the exact position x^* of the optimal highway. Let $x' \in X$ be the value that minimizes the expression in Equation (1). Note that the point pair $\{a, b\}$ in Lemma 2 is the pair $\{\alpha_{x'}, \beta_{x'}\}$. We also know the rough position of the optimal highway, i.e. we know that $x' \leq x^* \leq x''$, where x'' is the successor of x' in X.

It remains to find the pair $\{c, d\}$ with $(x^*, t^*) = s^2_{cd} \cap s^3_{ab}$. Note that this intersection point is rightmost among all intersection points of type $s^2_{pq} \cap s^3_{ab}$, where $\{p, q\}$ is any pair in S. (Can we use this fact? Seems unlikely since segments of type s^2_{pq} that do not intersect s^3_{ab} do not give us any information about where to continue the binary search.) Heekap's example shows that there may be a (linear?) number of pairs $\{p, q\}$ with $s^2_{pq} \cap s^3_{ab} \neq \emptyset$.

Recall that task (B) is the following: given a highway position x (with $x' \leq x \leq x''$ as we know now), find a pair $\{\gamma, \delta\}$ in S that is not separated by the current highway and whose travel-time distance is larger than $\mathrm{hw}_x(a, b)$—if such a pair exists. Consider the set $R^+_p(x)$ of points in h^+_x of distance at most $\ell = \mathrm{hw}_x(a, b)$ from a point $p \in h^+_x$. If $x_p - x \geq \ell$, then $R^+_p(x)$ is simply the L_1-circle C_p of radius ℓ centered at p, see Figure 6(a). If $0 \leq x_p - x < \ell$, then $R^+_p(x)$ is the union of a triangle $T_p(x)$ and the set $C_p \cap h^+_x$, see Figures 6(b) and (c). Let $\Delta x_p = \ell - (x_p - x)$. Then the triangle $T_p(x)$ connects the points $(x + \Delta x_p, y_p)$, $(x, y_p + v\Delta x_p)$, and $(x, y_p - v\Delta x_p)$. Let $R^-_p(x)$ be defined symmetrically for points $p \in h^-$. Now task (B) is equivalent to checking whether there are two points $\{\gamma, \delta\} \subseteq S^+_x$ such that $\gamma \notin R^+_\delta(x)$ or whether there are two points $\{\gamma, \delta\} \subseteq S^-_x$ such that $\gamma \notin R^-_\delta(x)$. We give an algorithm for checking the first condition (B^+), the second condition (B^-) is symmetric.

Recall that the subset S^+_x of S in h^+_x is the same for any highway position $x \in [x', x'']$. We also know that there is a highway position $x^* \in [x', x'']$ (namely the optimal one) such that $S^+_x \subset R^+_p(x^*)$ for each $p \in S^+_x$. The inclusion $S^+_x \subset R^+_p(x)$ actually holds for any $x \geq x^*$ since the regions of type $R^+_p(x)$ are monotonous in the sense that $R^+_p(x_1) \cap h^+_{x_2} \subset R^+_p(x_2)$ for $x_1 < x_2$. For each point $p \in S^+_x$ let f_p be a point in S^+_x such that if $f_p \in R^+_p(x)$ then $S^+_x \subset R^+_p(x)$. Note that either f_p is an arbitrary point in the L_1-circle C_p (if $S^+_x \subset C_p$) or f_p is the last point hit



(a) (b) (c) (d)

Figure 6: Examples of different regions of type R_p^+ for highway speed $v = 2$.

by the boundary $\partial R_p^+(x)$ of $R_p^+(x)$ while x increases. Let g_p be the horizontal line through p directed in positive x-direction. For points $q, r \in h^+ \cap g_p^-$ we say that r is *later* w.r.t. p than q if the line with slope $-v$ through r intersects g_p to the right of the line with the same slope through q. We extend this definition to points below p using lines of slope v, see Figure 7.

Now it is clear that condition (B^+) is equivalent to $f_p \in R_p^+(x)$ for all $p \in S_x^+$. If we knew all points of type f_p, then condition (B^+) could be checked in linear time. Note that the points f_p are independent of the exact position of the highway between x' and x''. In the remainder of this section we detail how to precompute the points f_p in $O(n \log n)$ time. This will complete the proof of Theorem 1.

First note that the set $\bigcup_{x \in [x', x'']} R_p^+(x)$ is contained in the rombus B_p of width 2ℓ and height $2v\ell$ that is obtained by scaling C_p vertically by a factor of v. If $B_p \setminus C_p$ is empty, then f_p is an arbitrary point in S_x^+, e.g. $f_p = p$. Otherwise f_p lies in a kite-shaped region above or below C_p. In Figure 6(d) the upper component K_p^{up} of $B_p \setminus C_p$ is shaded. We now show how to find the latest point $f(K_p^{\mathrm{up}})$ in that region (if any); the latest point $f(K_p^{\mathrm{lo}})$ in the lower component K_p^{lo} of $B_p \setminus C_p$ can be obtained in the same way. If both points exist, f_p is the later of them. Figure 8 shows how the kite K_p^{up} can be covered by two triangles Δ_p^{ul} and Δ_p^{ur} both whose upper right edges have slope $-v$. Note that this is the same slope as that of the top right edge of the triangle $T_p(x)$ that moves right with increasing x. Accordingly we split the task of computing $f(K_p^{\mathrm{up}})$ into that of computing the latest point $f(\Delta_p^{\mathrm{ul}})$ of Δ_p^{ul} and that of computing the latest point $f(\Delta_p^{\mathrm{ur}})$ of Δ_p^{ur}. If $f(\Delta_p^{\mathrm{ur}})$ is defined, i.e. if $\Delta_p^{\mathrm{ur}} \cap S_x^+ \neq \emptyset$, then $f(K_p^{\mathrm{up}}) = f(\Delta_p^{\mathrm{ur}})$, otherwise $f(K_p^{\mathrm{up}}) = f(\Delta_p^{\mathrm{ul}})$. Again we only detail how to determine $f(\Delta_p^{\mathrm{ur}})$, computing $f(\Delta_p^{\mathrm{ul}})$ is analogous.

Observe that the sets Δ_p^{ur} with $p \in S_x^+$ are translates of each other. In each set Δ_p^{ur} with $\Delta_p^{\mathrm{ur}} \cap S_x^+ \neq \emptyset$ we want to find the latest point $f(\Delta_p^{\mathrm{ur}}) \in \Delta_p^{\mathrm{ur}} \cap S_x^+$. Note that this is the point in $\Delta_p^{\mathrm{ur}} \cap S_x^+$ closest to the upper right edge of Δ_p^{ur}. The following lemma shows that we can find all points of type $f(\Delta_p^{\mathrm{ur}})$ in $O(n \log n)$ time as desired.

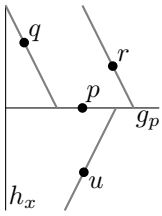


Figure 7: Point r is later than q and u.

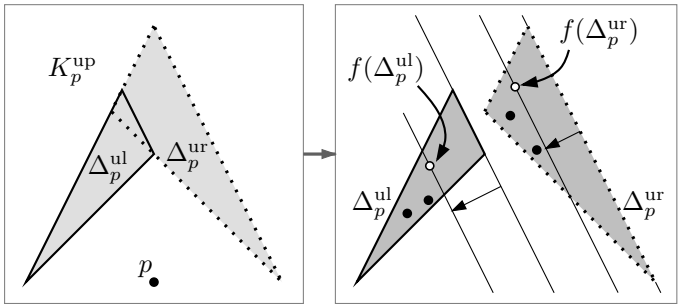


Figure 8: Splitting the task of determining $f(K_p^{\mathrm{up}})$ into two easier tasks.

Lemma 5 *Given a set P of n points, a set $\mathcal{T}$ of m triangles that are translates of each other, and for each triangle $T \in \mathcal{T}$ an edge e_T of T, we can compute in $O((n+m)\log m)$ time for each non-empty $T \in \mathcal{T}$ a point $f_T \in P \cap T$ that is closest to e_T.*

Proof. First note that it is enough to give an algorithm for the case that all edges e_T are translates of each other. If not, we simply run the algorithm for the restricted case three times, once for each class of edges. In our algorithm we sweep the plane with a line that is parallel to all edges of type e_T. The sweeping direction is such that for each $t \in \mathcal{T}$ the seep line hits e_T before v_T, the vertex of T opposite of e_T, see Figure 9. This ensures that in each non-empty triangle T the point f_T is hit by the sweep line first. We say that v_T is the *reference point* of T.

Before the sweep we set up a range-searching data structure $\mathcal{R}$ for the set $V_{\mathcal{T}} = \{v_T \mid T \in \mathcal{T}\}$ of reference points. Our ranges are translates of a triangle that is point-symmetric to the triangles in $\mathcal{T}$. After constructing $\mathcal{R}$ in $O(m \log m)$ time, queries can be answered in this setting in $O(k+\log m)$ time [CE87, AE99], where k is the number of points that are reported. Since the range-reporting data structure is based on point location (this works, but what do the references actually say?), it is no problem to delete points in $O(\log m)$ time.

The sweep line stops whenever it hits a point p in P. Then we query $\mathcal{R}$ with the triangle $\tau(p)$ that is obtained by the concatenation of two congruence mappings: translate any $T \in \mathcal{T}$ by the vector $p - v_T$ and then rotate the result around p by $180°$ degrees, see Figure 9. This query yields all points v_T with $p \in T$ that have not been removed before. After reporting these points, we remove them from $\mathcal{R}$.

The correctness of the algorithm relies on two facts. First, f_T is the first point in $P \cap T$ hit by the sweep line. Second, $\tau(p)$ is exactly the set of all points in the plane that are reference points of triangles containing p, see Figure 9.

The running time is easy to see. □

5 Extensions

- What about placing the optimal combination of a vertical and a horizontal highway?
- What about the Euclidian case?

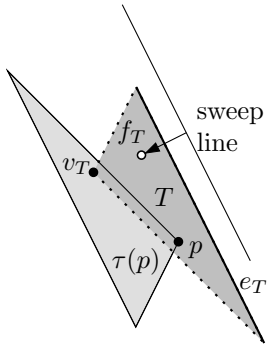


Figure 9: The point p is contained in all triangles T with reference point $v_T \in \tau(p)$.

References

[AE99] Pankaj K. Agarwal and Jeff Erickson. Geometric range searching and its relatives. In B. Chazelle, J. E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, volume 223 of *Contemporary Mathematics*, pages 1–56. American Mathematical Society, Providence, RI, 1999.

[CE87] Bernard Chazelle and H. Edelsbrunner. Linear space data structures for two types of range search. *Discrete Comput. Geom.*, 2:113–126, 1987.